

Delphi e DLLs

Como produzir e utilizar DLLs em seus aplicativos Delphi

Normalmente um programador possui várias funções que são utilizadas em vários aplicativos. Qual o melhor lugar para armazená-los? E como posso fazer uso de rotinas que a linguagem de programação que utilizo não permite (como no Visual Basic)?

A solução para esses problemas é o uso de DLLs (Dinamic Link Libraries). Um DLL é um programa que permite que vários programas possam acessar as mesmas funções, independentemente de linguagem.

Criando DLLs:

Primeiramente vá em File|New... e escolha DLL. Surgirá na tela um “esqueleto” de um programa DLL:

```
library Project1;
```

```
{ Important note about DLL memory management: ShareMem must be the  
first unit in your library's USES clause AND your project's (select  
View-Project Source) USES clause if your DLL exports any procedures or  
functions that pass strings as parameters or function results. This  
applies to all strings passed to and from your DLL--even those that  
are nested in records and classes. ShareMem is the interface unit to  
the DELPHIMM.DLL shared memory manager, which must be deployed along  
with your DLL. To avoid using DELPHIMM.DLL, pass string information  
using PChar or ShortString parameters. }
```

```
uses
```

```
    SysUtils,  
    Classes;
```

```
begin
```

```
end.
```

Para que uma procedure ou função possam ser utilizados, é necessário informar que eles podem ser exportados (utilizados por outros aplicativos), colocando-se as diretivas **stdcall**; **export**; após sua declaração. No final do programa (antes do **begin - end.**), deve-se informar as rotinas a serem exportadas e seus respectivos índices (pode ser qualquer um, desde que seja único em todo o DLL). Caso não seja informado não poderá ser utilizado por outros programas.

Um exemplo completo de criação de DLL:

```
library Exemplo;
```

```
uses
```

```

SysUtils,
Classes;

function AreaQuadrado (Lado : Integer) : Integer; stdcall; export;
begin
    Result := Lado * Lado;
end;

function AreaTrapezio (lMaior, lMenor, Altura : Integer) : Extended; stdcall; export;
begin
    Result := ((lMaior + lMenor) * Altura) / 2;
end;

function AreaCirculo (Raio : Integer) : Extended; stdcall; export;
begin
    Result := 3.14159 * Raio;
end;

exports
    AreaQuadrado index 1,
    AreaTrapezio index 2,
    AreaCirculo index 3;

begin
end.

```

Utilizando DLLs em Aplicativos Delphi

Há dois métodos de colocar um DLL em um programa: o método estático e o dinâmico. A diferença entre eles é que no primeiro caso, o DLL é carregado com o programa, e não roda sem ele. Ou seja, caso o DLL não esteja no diretório do programa ou no diretório do Windows, o programa não executa. No segundo método, o DLL é executado durante a execução do programa, e pode ser liberado da memória logo após a execução. Usar o melhor método é uma questão de escolha.

Método Estático:

implementation

```
function AreaQuadrado(Lado : Integer) : Integer; external 'exemplo.dll' index 1;
```

A declaração de uma função ou procedure externa é praticamente idêntica à declaração de uma função “comum”, as únicas diferenças são:

- Não é necessário incluir a declaração na seção **interface**, somente na **implementation**;
- Não é necessário colocar **begin - end** na função, simplesmente declará-la é suficiente.

Método Dinâmico:

interface

```

type
  AreaQuadrado = function (Lado : Integer) : Integer;

implementation

procedure TfrmMain.btAreaClick (Sender : Tobject);
var
  Area : Integer;
  Handle : THandle;
begin
  Handle := LoadLibrary ('exemplo.dll');
  if Handle <> 0 then
    begin
      @AreaQuadrado := GetProcAddress (Handle, 'AreaQuadrado');
      if @AreaQuadrado <> nil then
        begin
          Area := AreaQuadrado (strtoint (edArea.Text));
          ShowMessage ('A área do quadrado é de ' + inttostr (Area) + 'm2 ');
        end;
      FreeLibrary (Handle);
    end;
  end;

```

Observações sobre DLLs:

- Nunca faça acesso a dados confidenciais em DLLs (como esquemas de senha), mesmo que precise ser usado em várias linguagens. É mais fácil, porém menos seguro (pode ser que algum cracker - usar hacker como pirata de computador é um erro - descubra o índice do DLL e faça uso indevido do programa);
- Tome cuidado com o tipo String. O Delphi implementa o uso de strings longas, mas pode ser que outras linguagens não a implementem. Portanto é recomendável o uso de ShortString ou PChar como tipo de retorno em funções e em parâmetros de Strings;
- Sempre trate qualquer tipo de erro a partir do DLL: caso ela não seja tratada, o erro vai “vazar” para o programa, e o resultado é impossível de se dizer com precisão;
- Lembre-se de sempre remover os DLLs da memória (quando for fazer acessos dinâmicos);
- Nunca se esqueça de colocar o seu DLL em um dos seguintes diretórios: o diretório do programa, o diretório do Windows (normalmente C:\Windows), o diretório de sistema do Windows (normalmente C:\Windows\System) ou qualquer diretório que esteja na cláusula “PATH” do arquivo AutoExec.Bat.